CREDIT-BASED FLOW CONTROL OVER UNRELIABLE LINKS

BACKGROUND OF THE INVENTION

A. Field of the Invention

[0001] The present invention relates generally to data switching and routing, and more particularly, to data flow control over a switch fabric.

B. Description of Related Art

[0002] Routers receive data on a physical media, such as optical fiber, analyze the data to determine its destination, and output the data on a physical media in accordance with the destination. Routers were initially designed using a general purpose processor executing large software programs. As line rates and traffic volume increased, however, general purpose processors could not scale to meet these new demands. For example, as functionality was added to the software, such as accounting and policing functionality, these routers suffered performance degradation. In some instances, the routers failed to handle traffic at line rate when the new functionality was turned on.

[0003] To meet the new demands, purpose-built routers were architected. Purpose-built routers are designed and built with components optimized for routing. They not only handled higher line rates and higher network traffic volume, they also added functionality without compromising line rate performance.

[0004] A purpose-built router may include a number of input and output ports from which it transmits and receives information packets. A switching fabric may be implemented in the router to carry the packets between ports.

[0005] Flow-control refers to the metering of packet flow through the network and/or through the router. For example, it may be desirable to limit the number of packets transmitted from a particular port of the router in order to ensure that the router's switching fabric is not overloaded. One known method of implementing flow-control controls flow on a per-queue basis. One disadvantage of per-queue flow control is that it may not evenly distribute flow across all elements of the system. For example, in a system using a switch fabric, the fabric may become congested even though there is per-queue flow control.

[0006] It is therefore desirable to efficiently implement flow control without the performance degradation caused by conventional per-queue controls.

<u>SUMMARY OF THE INVENTION</u>

[0007] Systems and methods consistent with the invention address, among other things, a switch-fabric credit-based flow control technique that compensates for lost packets.

[0008] One aspect of the principles of the invention is directed to a network device connected to a switching fabric. The network device includes a credit counter configured to store a value indicating an amount of data eligible to be transmitted from the network device. For example, the network device may decrement the credit counter when it transmits a packet to the switching fabric. If

the credit counter is decremented below a predetermined level, the port refrains from transmitting packets. The credit counter is replenished (incremented) based on, for example, a timer or an increment signal received from the switching fabric. If the signal is lost in transmission, however, the credit will be lost and the credit counter will not be replenished. Lost credits can result in under utilization of router resources leading to performance degradation. To address this issue, the network device also includes a request component and a fake request circuit. The request component generates requests to send data to the switching fabric and receives corresponding grants in response to the generated requests. The request component decrements the credit counter when the requests are generated and increments the credit counter when the corresponding grants are received. The fake request circuit generates fake requests that cause grants to be returned to the requesting component that increment and thus replenish the counter. When generating the fake requests, the credit counter is not decremented.

[0009] A second aspect of the principles of the invention is directed to a request controller for metering data flow to a network. The request controller includes a real request vector component and a fake request vector component. The real request vector component generates request messages corresponding to data units that are to be transmitted to the network and receives back grant messages indicating that the data units can be transmitted to the network. The fake request vector component periodically generates a fake request message to a destination on the network determined by a value in a pointer register. The

pointer register is incremented after each of the fake request messages are generated.

[0010] A third aspect of the principles of the invention is directed to a method of metering data flow to a network. The method includes receiving at least one data unit, generating a request to transmit the data unit when a credit counter contains sufficient credits for the data unit, and decrementing the credit counter in response to generating the request to transmit the data unit.

[0011] Additionally, the method includes receiving grant messages that correspond to the transmitted requests and incrementing the credit counter in response to the grant messages. Periodically, a fake request is generated that does not correspond to a data unit. The fake request causes grant messages to be received from the network and the credit counter to be incremented in response thereto.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate an embodiment of the invention and, together with the description, explain the invention. In the drawings,

[0013] Fig. 1 is a block diagram illustrating a routing system consistent with the principles of the invention;

[0014] Fig. 2 is a block diagram illustrating portions of the routing system shown in Fig. 1 in additional detail;

[0015] Fig. 3 is a diagram illustrating an implementation of a communication component shown in Fig. 2;

[0016] Fig. 4 is a diagram illustrating the fabric request controller shown in Fig. 3 in additional detail;

[0017] Fig. 5A is a flow chart illustrating operation of the fabric request controller in transmitting requests;

[0018] Fig. 5B is a flow chart illustrating operation of the fabric request controller in receiving grants;

[0019] Fig. 6 is a diagram illustrating the fake request vector component shown in Fig. 4 in additional detail; and

[0020] Fig. 7 is a flow chart illustrating a method of operation of the fake request vector component consistent with an embodiment of the present invention.

DETAILED DESCRIPTION

[0021] The following detailed description of the invention refers to the accompanying drawings. The same reference numbers may be used in different drawings to identify the same or similar elements. Also, the following detailed description does not limit the invention. Instead, the scope of the invention is defined by the appended claims and equivalents.

[0022] As described herein, credit based flow-control is implemented using a request-grant credit scheme. A credit counter is decremented whenever a request to transmit information is issued and incremented whenever a

corresponding grant to transmit the information is received. Extra credits are occasionally added to the credit counter based on the average rate of corruption on the switching fabric. The extra credits are added indirectly through the generation of extra requests. In this manner, lost credits due to transmission errors can be avoided.

SYSTEM DESCRIPTION

[0023] Fig. 1 is a block diagram illustrating a routing system 42 consistent with the principles of the invention. System 42 includes packet forwarding engines (PFEs) 44, 46..., and 48, a switch fabric 50, and a routing engine (RE) 52. System 42 receives a data stream from a physical link, processes the data stream to determine destination information, and transmits the data stream out on a link in accordance with the destination information.

[0024] RE 52 performs high level management functions for system 42. For example, RE 52 communicates with other networks and systems connected to system 42 to exchange information regarding network topology. RE 52 creates routing tables based on the network topology information and forwards the routing tables to PFEs 44, 46, . . . , and 48. The PFEs use the routing tables to perform route lookup for incoming packets. RE 52 also performs other general control and monitoring functions for system 42.

[0025] PFEs 44, 46, . . . , and 48 are each connected to routing engine (RE) 52 and switch fabric 50. PFEs 44, 46 . . ., and 48 receive data on ports connecting physical links connected to a wide area network (WAN). Each physical link could be one of many types of transport media, such as optical fiber

or Ethernet cable. The data on the physical link is formatted according to one of several protocols, such as the synchronous optical network (SONET) standard, an asynchronous transfer mode (ATM) technology, or Ethernet.

[0026] PFE 44 will be used to discuss the operations performed by a PFE consistent with the principles of the invention. PFE 44 processes incoming data by stripping off the data link layer. PFE 44 converts the remaining data into data structures called D cells.

[0027] For example, in one embodiment the data remaining after the data link layer is stripped off is packet data. PFE 44 stores the layer 2 (L2) and layer 3 (L3) packet header information, some control information regarding the packets, and the packet data in a series of D cells. In one embodiment, the L2, L3, and control information are stored in the first two cells of the series.

[0028] PFE 44 forms a notification based on the L2, L3, and control information and performs a route lookup using the notification and the routing table from RE 52 to determine destination information. PFE 44 may also process the notification to perform protocol-specific functions, policing, and accounting, and might even modify the notification to form a new notification.

[0029] If the determined destination indicates that the packet should be sent out on a physical link connected to PFE 44, then PFE 44 retrieves the cells for the packet, converts the new notification into header information, forms a packet using the packet data from the cells and the header information, and transmits the packet from the port associated with the physical link.

[0030] If the destination indicates that the packet should be sent to another PFE via switch fabric 50, then PFE 44 retrieves the cells for the packet, modifies the first two cells with the new notification and new control information, if any, and sends the cells to the other PFE via switch fabric 50. The receiving PFE uses the notification to form a packet using the packet data from the cells, and sends the packet out on the port associated with the appropriate physical link of the receiving PFE.

[0031] In summary, in one embodiment, RE 52, PFEs 44, 46, and 48 and switch fabric 50 perform routing based on packet-level processing. PFEs store each packet using cells while performing a route lookup using a notification, which is based on packet header information. A packet might come in from the network on one PFE and go back out to the network on the same PFE, or be sent through switch fabric 50 to be sent out to the network on a different PFE.

[0032] Fig. 2 is a block diagram illustrating portions of routing system 42 in additional detail. PFEs 44, 46, and 48 connect to one another through switch fabric 50. Each of the PFEs may include one or more physical interface cards (PICs) 201-202 and flexible port concentrators (FPCs) 205.

[0033] PICs 201-202 transmit data between a WAN physical link and FPC 205. PICs are designed to handle different types of WAN physical links. For example, PIC 201 may be an interface for an optical link while PIC 202 may be an interface for an Ethernet link. Although Fig. 2 shows two PICs 201 and 202 connected to the FPCs 205, in other embodiments consistent with the invention there can be a single PIC or more than two PICs connected to an FPC 205.

[0034] Switch fabric 50 includes switches 220 that transmit cells through the fabric 50. The switches may be connected via optical links and may be organized into multiple fabric planes 230. In one embodiment, four fabric planes 230 are used.

[0035] FPCs, such as FPC 205, handles packet transfers to and from PICs 201 and 202, and switch fabric 50. For each packet it handles, FPC 205 performs the above-discussed route lookup function. FPCs 205 communicate with switch fabric 50 through a fabric communication component 207 (labeled as Nout). Communication component 207 handles the protocols associated with physically transmitting and receiving cells with switch fabric 50.

[0036] Fig. 3 is a block diagram illustrating an implementation consistent with the principles of the invention for one of communication components 207. Communication component 207 comprises a notification queue manager 305, a notification buffer pool 306, a grant pending queue 307, a fabric request controller 308, a packet reader 309, and a data buffer 310. Fabric request controller 308 additionally includes a credit-counter (cc) 320 used to implement flow-control.

[0037] Data transmission from communication component 207 begins when notification queue manager 305 receives a notification, signifying that notification and data cells are to be sent to another FPC. Upon receiving the notification, notification queue manager 305 stores the notification in notification buffer pool 306. In response, notification buffer pool 306 returns an address defining where the notification is stored in notification buffer pool 306.

Notification queue manager 305 stores the received address in one or more internal queues. Notification queue manager 305 arbitrates across its internal notification address queues to select a notification for processing. Selected notifications are sent to grant pending queue 307 from notification buffer pool 306.

[0038] In order for communication component 207 to transmit the notification and its associated data cells to switch fabric 50, grant pending queue 307 firsts requests permission, via a request signal to fabric request controller 308. More particularly, grant pending queue 307, when it receives a notification, sends a request, that includes the destination and the number of cells in the packet. Grant pending queue 307 holds the outgoing notification until permission, called a grant, to send the packet data cells associated with the notification is received back from fabric request controller 308. When permission is granted, grant pending queue sends the notification to packet reader 309, which forwards the cells to data buffer 310 for transmission over the switch fabric 50.

[0039] In an embodiment consistent with the present invention, fabric request controller 308 uses credit counter 320 to implement credit-based flow control for the requests received from grant pending queue 307. The credit-based flow control is implemented in the context of an unreliable switch fabric (i.e., unreliable links). A technique for implementing credit-based flow control in the context of reliable links is disclosed in U.S. Patent Application Serial Number

09/448,124, by Phillippe G Lacroute, et al., filed November 24, 1999 and titled "Switching Device," the contents of which are hereby incorporated by reference.

[0040] A more detailed description of the operation of fabric request controller 308 in the context of an unreliable switch fabric will now be described with reference to Figs. 4-7.

OPERATION AND IMPLEMENTATION OF FABRIC REQUEST CONTROLLER

[0041] Fig. 4 is a block diagram illustrating an embodiment of fabric request controller 308 consistent with the principles of the invention. Fabric request controller 308 includes a real request vector component 401 and a fake request vector component 402. Each of components 401 and 402 generate a request vector that indicates to which of the possible destinations a request should be sent. In one implementation, switch fabric 50 may interconnect 144 devices, such as 144 FPCs. Accordingly, the request vector may be stored in a 144-bit register, with each bit corresponding to one of the possible destinations.

[0042] Arbiter 403 receives the request vectors from real request vector component 401 and fake request vector component 402, combines the two request vectors and transmits requests to each destination indicated by the combined request vectors. In one embodiment arbiter 403 is a round robin arbiter. Arbiter 403 may, for example, logical OR the two request vectors to obtain the combined request vector.

[0043] Figs. 5A and 5B are flow charts illustrating operation of fabric request controller 308 in transmitting request messages and receiving back grant messages corresponding to the requests.

[0044] Fig. 5A illustrates an exemplary method for transmitting requests. To begin, for each request received from grant pending queue 307, real request vector component 401 checks if the credit counter 320 contains sufficient credits (Act 501). If not, the request is queued until credits are available (Act 502). If credits are available, the real request vector component 401 decrements the credit counter 320 and sets the real request vector to indicate the destination(s) with which the request is associated (Acts 503 and 504). As previously mentioned, the real request vector may be a 144-bit register, where each bit corresponds to a possible destination. For example, if the data is to be sent to the destinations associated with the fifth and tenth bits in the request vector, real request vector component 401 would accordingly set the fifth and tenth bits in the real request vector to logical one.

[0045] Fake request vector component 402 similarly creates a 144-bit "fake" request vector (Act 505). Periodically, arbiter 403 reads the two request vectors and combines them into a single request vector (Act 506). Arbiter 403 transmits a request to switch fabric 50 for each destination set in the combined request vector (Act 507).

[0046] A grant message indicates that the data cells associated with a request can be sent from the FPC 205. Fig. 5B illustrates an exemplary method for receiving a grant from switch fabric 50 in response to a previously sent

request. If the credit counter 320 is not saturated (i.e., it is not at its maximum value), the credit counter is incremented. (Acts 510 and 511). Fabric request controller 308 then signals grant pending queue 307 that the requested cells can be sent to switch fabric 50 via packet reader 309 and data buffer 310. (Act 512).

[0047] Fabric request vector component 402 generates "fake" request vectors that do not correspond to data cells in grant pending queue 307. The fake requests compensate for real requests that are periodically lost in the switch fabric 50. Fig. 6 is a diagram illustrating an embodiment of fake request vector component 402. Component 402 includes a timing counter 601, a programmable register 602, a comparator 603, a pointer 604, vector setting circuit 615, and a fake request vector 605. In one embodiment, fake request vector 605 is a 144-bit register and pointer 604 is an eight bit value that stores a reference to a location in fake request vector 605.

[0048] Timing counter 601, programmable register 602, and comparator 603 operate in conjunction with one another to generate a periodic signal that triggers the setting of fake request vector 605 by vector setting circuit 615. Counter 601 increases its count value at a rate determined by an input clock signal. Comparator 603 compares the value in counter 601 with the value in programmable register 602. When the value in counter 601 equals or exceeds the value in programmable register 602, comparator 603 generates signal 611, which resets counter 601 and activates vector setting circuit 615. In response, vector setting circuit 615 sets the bit in fake vector 605 that is referenced by pointer 604.

[0049] Fig. 7 is a flow chart illustrating a method of operation of fake request vector component 402 consistent with an embodiment of the invention. In this embodiment, counter 601 is a 24-bit counter, programmable register 602 is a 21-bit register, pointer 604 is an eight-bit register, and fake request vector 605 is a 144-bit register. These components could be substituted for components of other sizes in other embodiments.

[0050] In general, fake request vector component 402 generates fake requests at a rate that tends to compensate for real requests to switch fabric 50 that never receive back a corresponding grant signal. The rate can be determined in a variety of ways. For example, in one embodiment, the average rate of lost requests is empirically measured for the router. Lost requests may be caused by, for example, errors occurring in the optical transmission links in switch fabric 50. The credit counter 320 is not decremented when a fake request is issued, although it is incremented when a corresponding grant is received back from a destination. Real request vector component 401 does not distinguish between grants corresponding to real requests or fake requests. Accordingly, fake request vector 402 compensates for lost credits in a manner that is transparent to the circuitry in real request vector component 401 associated with the credit counter.

[0051] To begin, programmable register 602 is set to a preset value by the user (Act 701). The appropriate preset value to use may be based on an empirical observation of the error rate of the system and on the frequency of the clock driving counter 601. For example, in one embodiment, the programmable

register 602 is set to a value that produces an interval of 7.4 micro-seconds to 500 milli-seconds. The upper 21 bits of the counter 601 is compared to the value in the programmable register 602 (Act 702). When the count value is greater than the value in the programmable register 602, the counter is cleared, via signal 611, and begins counting anew (Act 703). Additionally, vector setting circuit 615 sets the bit in fake vector register 605 at the location identified by pointer 604 and increments the pointer to point to the next position in fake vector register 605 (Acts 704 and 705). Because an eight-bit pointer can contain more than 144 values, vector setting circuit 615 also checks whether pointer 604 is greater than 143, and if so, resets the pointer to zero (Acts 706 and 707).

[0052] As described above, a fake request vector component periodically generates fake requests to a network fabric in order to compensate for real requests that are lost in the network fabric. The fake request vector component has a relatively simple hardware implementation that can generate fake requests at a high speed and programmable rate without interrupting data flow.

[0053] Although described in the context of a purpose-built router, concepts consistent with the invention can be implemented in any system that directs traffic flow based on a credit counter that is used to control data transmission resources based on a request-grant scheme.

[0054] The foregoing description of preferred embodiments of the present invention provides illustration and description, but is not intended to be exhaustive or to limit the invention to the precise form disclosed. Modifications

and variations are possible in light of the above teachings or may be acquired from practice of the invention.

[0055] The scope of the invention is defined by the claims and their equivalents.